A CONCEPTUAL MODEL FOR INTEGRATED AUTONOMOUS PROCESSING: AN INTERNATIONAL BANK'S EXPERIENCE WITH LARGE DATABASES

William F. Frank

Stuart E. Madnick

Y. Richard Wang

March 1987 #WP 1866-87

A CONCEPTUAL MODEL FOR INTEGRATED AUTONOMOUS PROCESSING: AN INTERNATIONAL BANK'S EXPERIENCE WITH LARGE DATABASES

William F. Frank

Stuart E. Madnick

Y. Richard Wang

March 1987 #WP 1866-87

A Conceptual Model for Integrated Autonomous Processing: An International Bank's Experience with Large Databases

William F. Frank
Integrated Information Systems Associates
Warren, Vermont

Stuart E. Madnick
Sloan School of Management
Massachusetts Institute of Technology

Y. Richard Wang
Department of Management Information Systems
College of Business and Public Administration
University of Arizona

Working Paper # 1866-87
Sloan School of Management, M.I.T.

Report #CIS-87-04
Composite Information Systems Project

TABLE OF CONTENTS

1. BANKING ENVIRONMENT

The banking environment has experienced dramatic changes over the past two decades and it continues to change at an accelerated pace. Competitive pressure is increasingly being imposed on the banking industry on a multitude of fronts. There is little to constrain wholesale oriented institutions from crossing interstate barriers and encroaching on local banks' territory. Competition is also crossing international frontiers; in many countries international wholesale banking is becoming much more aggressive as banks vie for the business of multinational corporations.

Information technology is being used extensively in banking [Lipis, 1985]. For example, in 1986 the value of computerized payments processed by the New York Federal Reserve and New York Clearing House often exceeded $1 trillion in a single day. Technology has been critical to keep pace with the increased volume of financial activities; payments processed through New York financial institutions have increased 50-fold in the past 20 years, to the point where every four days an amount equal to the total annual US GNP is turned over.

During the 1990's, technological innovation will have an even more dominant effect on the financial services environment than interest rate volatility. The development of sophisticated information processing facilities are enabling institutions to offer a much broader range of products and services on a global basis with great efficiency. Technologies such as distributed systems and database machines [Hsiao and Madnick, 1977; Madnick, 1977] are being employed by the industry to gain strategic advantage [Keen, 1986; McFarlan, 1984].

1.1 AUTONOMY, INTEGRATION, AND EVOLUTION

This paper reviews and analyzes the development and deployment of a conceptual model for integrated autonomous processing in a major international bank. The cultural forces in the bank favored an autonomous non-integrated approach: a tradition in which the responsibility for getting a job done is itself distributed to the lowest possible level, and in which the independence of projects was thus always maximized. To cope with the constantly changing environment and the distributed autonomous culture, it was recognized that three key goals need to be satisfied in designing information systems: autonomy, integration, and evolution.

Each bank product (e.g., funds transfer, letter of credit, loans, cash management) is developed <u>autonomously</u> by separate component personnel. Each product manager, in general, has complete freedom over his hardware acquisition and software development choices (e.g., hire his own development staff, retain outside contract programmers, or buy/modify suitable application packages). In the culture of this bank, this autonomy is critical since each manager is held solely responsible for his products: excuses such as "the data processing people didn't do it right" is not acceptable. When information must be exchanged, it was usually accomplished by "tape hand-offs", usually at night, as depicted in Figure 1.

On the other hand, the needs for <u>integration</u> have been increasing rapidly both at the user level and database level. Since each system had its own directly connected terminals, users that required access to multiple systems had to have multiple terminals in their office, or walk to an area of the building that had a terminal tied to the system

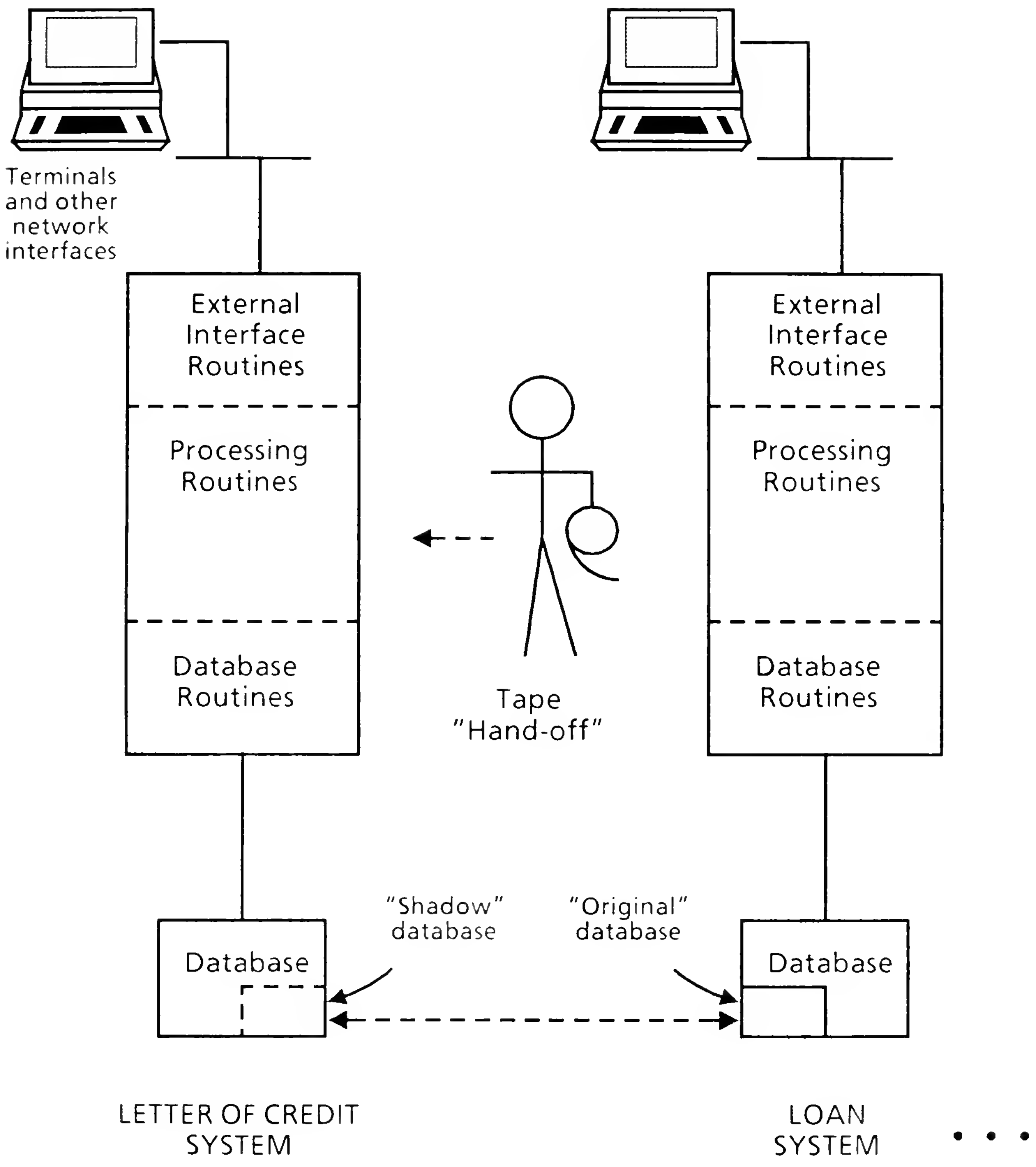


Figure 1.
Independent Autonomous Systems

needed. The tape hand-offs were used to create "shadow" databases of each other's real databases. Since the shadow database diverges from the real database during the day, inconsistencies could result.

The problem of integration has been intensified by the need for evolution in at least three areas: current products, new products, and new technology. As the current products become more sophisticated, there is need to acquire more information from the other systems. Increasing tape hand-offs lead to processing complexities and do not address the need for up-to-date information. Many of the new products (e.g., cash management) are, in fact, a repackaging and combination of existing products -- to produce completely new systems would be expensive, time consuming, and impractical due to a dramatic increase in requirements for tape hand-off information sharing. Finally, to maintain an efficient and cost-effective environment, it is important to be able to take advantage of new hardware components without disrupting or discarding existing systems.

Traditional centralized database management system strategies provide integration, but have limited capabilities for evolution and reduce managerial autonomy. The purpose of this paper is twofold: 1) present a conceptual model, based upon [Lam and Madnick, 1981; Madnick and Wang, 1987], that describes the architecture for an evolutionary, integrated, and autonomous environment; and 2) report the experience to date in implementing this model by the institutional banking group of a major international bank.

During the past three years, five products/services have been implemented using this conceptual model. This paper focuses on two of these systems: transaction investigations and management information system (MIS) reporting.

1.2 TRANSACTION INVESTIGATIONS

Complex international financial transactions performed in high volume can be error prone, resulting in a significant number of discrepancies between customers' records and bank records. Customers inquire about the source of these discrepencies and the bank is responsible for reviewing its records to establish what it did and why, supplying these records to the customer, and making good if the bank is in error. The bank's activity with respect to such an inquiry is called an investigation.

Investigations are the job of sizable staffs in a large bank. They are usually performed with the aid of data stored on the transaction processing systems, microfiche, and printed reports. In addition, the investigation activity itself generates a history, which should be tracked to allow indicators of productivity, customer satisfaction, and the efficacy of the transaction processing systems. The Historical Data Base (HDB), needed by the bank to support these investigations and other applications covering the previous 90 days, must be able to hold at least 40 million records.

1.3 MIS APPLICATIONS

Each transaction processing system generates various summary reports. Periodically, integrated MIS reporting for upper management was accomplished by manually entering data from a variety of these reports and other sources into an Apple computer spreadsheet.

Management desired new and more comprehensive MIS systems which integrated previously independently generated and inconsistently presented data. In total, a database of at least 12 million records, with over 250,000 additions per day, is needed to hold the relevant data. Furthermore, these new systems should facilitate the modelling of the effects of alternative decisions (i.e., what-if analyses fed by current real performance data).

2. CONCEPTUAL MODEL

2.1 MODEL ARCHITECTURE

The model developed consists of seven major functional components as depicted in Figure 2. These components are separated into five layers, with the integrating application-independent layers (external interface, message control, data control, and shared data resource) surrounding the application processing components [Madnick and Wang, 1987]. For this bank, these application processing components are separated into three classes of applications: transaction processing, information processing, and administrative processing.

2.2 AUTONOMY ASPECTS OF MODEL

This architecture attempts to mediate the conflicts between the goal of autonomy and the goals of integration and evolution. All of the layers, except for message control and data control, lend themselves to unlimited autonomy. Each product manager could acquire and manage his own resources including 1) terminal/network gateway

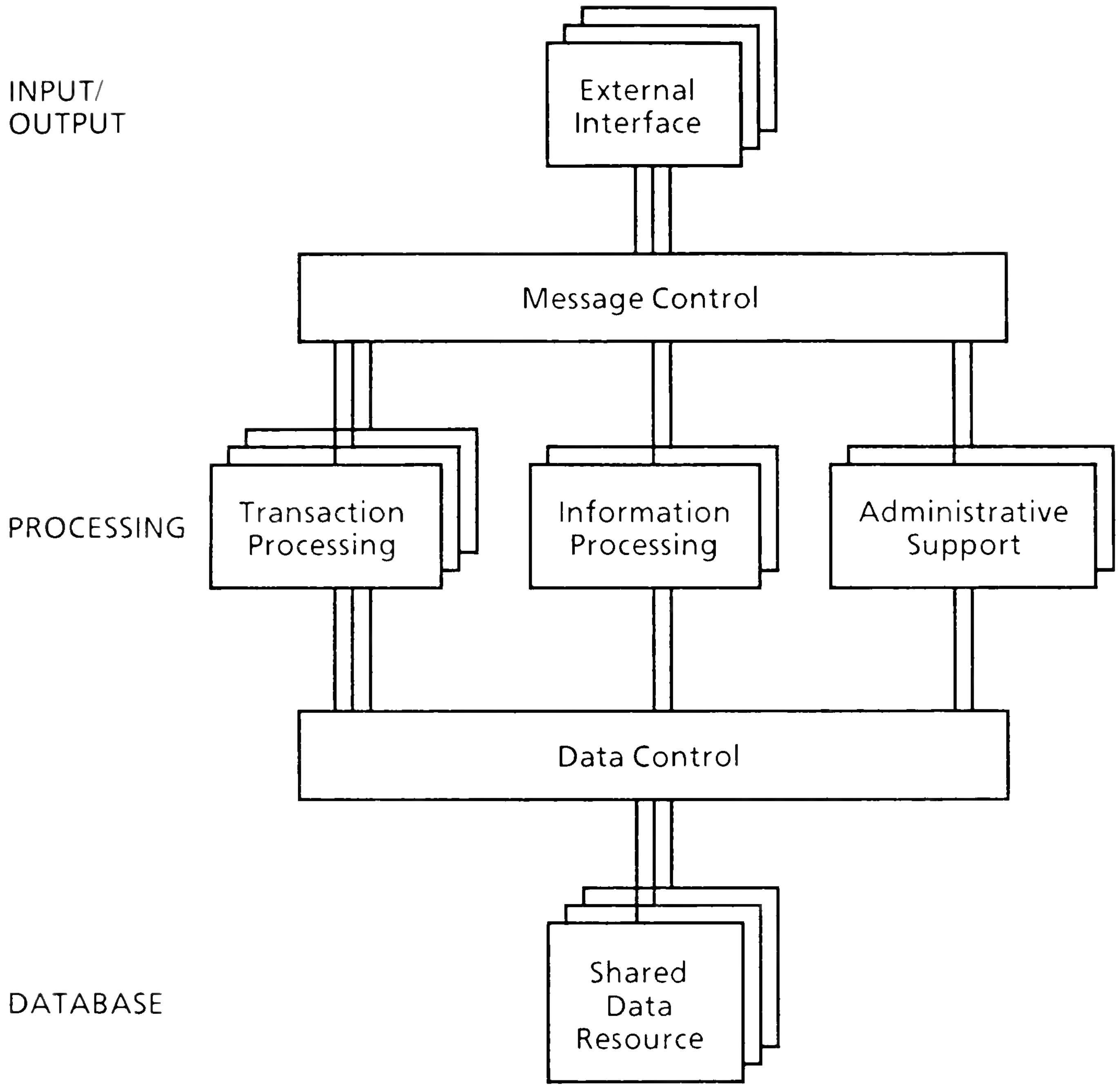


Figure 2.
Conceptual Model

hardware, 2) application processing computers and software, and 3) database computers and software.

In the past, as shown in Figure 1 earlier, these three decisions were bundled together. In practice, the primary concern for autonomy involved the application processing, with a lesser concern for the database, and minimal concern for the external interface. It is in the application processing that the functionality of the product is manifest. It is important that enhancements and corrections, as well as the initial development, be able to proceed with minimal needs for coordination or delays due to the managers of other areas of the bank and other computer systems.

Given the architecture in Figure 2, each manager has complete control over his application processing system. Furthermore, as many separate database systems as needed, or desired, can be created. It is expected that initially there will be more databases than theoretically needed, due to the influence of past practice. The integrated autonomous architecture provides access to these databases by other applications as well as an evolutionary path for eventually integrating these databases, as the needs for integration intensify.

2.3 INTEGRATION AND EVOLUTION ASPECTS OF MODEL

There are several underlying concepts and components of the model which address the issues of integration and evolution.

Message Control and Data Control perform unifying functions for the model. They are the points at which all processing will be coordinated. For example, in principle, any terminal can access any application. Furthermore, application subsystems can utilize the

Shared Data Resource to manage and maintain data which is common to more than one component.

Message Control and Data Control are both conceptually single entities. The other components are types of processing functions. There may be many instances of each type (e.g., multiple transaction processing systems and multiple shared data resources).

2.4 MODEL COMPONENTS

It is important to realize that the model components of Figure 2 are logically separate. They could be mapped to actual hardware in various ways. For instance, in the transaction investigation system, each component resides on a separate processor; whereas, in the MIS system, many components reside on a single processor.

2.4.1 External Interface

The external entities interfacing with these banking systems fall into five categories: 1) payment networks, 2) communication networks, 3) customer terminals, 4) professional workstations, and 5) other intra-bank and/or inter-bank systems.

2.4.2 Message Control

Message Control coordinates the passage of messages between the processing components. This involves routing, translation, sequencing, and monitoring:

- o Routing accepts a request for the delivery of messages to a particular logical function and determines the appropriate physical address to which the message should be delivered. Routing can thus accommodate changes in the availability and location of functions.

Routing can also help coordinate requests for services that involve several functions.

o Translation maps a limited number of protocols from one standard to another.

o Sequencing determines the order in which messages are to be delivered to recipients on the basis of established priorities.

o Monitoring determines the state of messages within the system at any given time. Monitoring thus includes the responsibility for the integrity of a message from the time it is presented by one component until it is accepted by another.

2.4.3 Transaction Processing

Transaction Processing refers to the applications which execute the customer's financial instructions. These systems typically retrieve and update a significant amount of data (e.g., client balances). The sub-functions of Transaction Processing include validation, risk management, accounting, and recording.

o Validation functions are those which perform review of instructions to ensure that all information needed for processing is present and that the information is consistent with previously defined rules for data elements. Incomplete or invalid requests may be "repaired" by augmenting or clarifying the request information, either with information available internally or from external sources such as the requestor.

o Risk Management functions are those which verify that the transactions being processed do not violate limits, conditions, or policies established by the customer, the bank, or the various regulatory agencies. Ideally, these functions should be synchronized with the processing of transactions. In some cases where this is not feasible, "after-the-fact" Risk Management functions can be used to initiate corrective action.

- o Accounting records the cumulative impact of the transactions completed. Accounting functions are characterized as being continuous over time, in contrast to the discrete events which take place in most of the transaction processing functions. For example, in the banking environment, accounting takes place on two distinct levels: customer accounting and organizational accounting.

2.4.4 Information Processing

Information Processing refers to all the subsystems that perform analysis, calculations, or restructuring of the data (e.g., consolidated financial statement). The sub-functions of Information Processing include user interface, static reporting, ad hoc reporting, and access to outside data resources.

2.4.5 Administrative Support

Administrative Support provides facilities for the performance of office functions by administrative or managerial personnel. This activity is required to maintain organization, procedural or personal information. Example facilities include electronic mail, word processing, correspondence files, and inventory controls. The sub-functions of Administrative Support provide

- o Template Presentation for the preparation of predefined documents.
- o Editing and Formatting for the alteration and preparation of documents and for automated checking of spelling and style.
- o Mail Delivery and Storage for the transmission of documents from one system to another.

- o Document Storage and Retrieval for maintaining lists, documents, tables, and passages from documents in an organization that allows retrieval on the basis of any of their attributes and relationships.
- o Control Functions to monitor administrative activities by maintaining task related information including status, performance, and activity data.
- o Graphic Functions for the preparation of diagrams, flow charts, organization charts, and other illustrative communication tools.

2.4.6 Data Control

Data Control coordinates access, presentation, and the passage of data between processing functions and the Shared Data Resources. It routes queries and updates to the appropriate component of the Shared Data Resources, performs security and priority functions, maintains concurrency control over the shared data, and returns responses to the appropriate processing function. Data Control must perform the following functions:

- o Security ensures that there is no unauthorized access to the Shared Data Resource and controls the view of the data permitted to different users.
- o Presentation provides standard, flexible and simple means for making query updates and data definition requests. Therefore, Data Control contains data manipulation and data definition language processing functions.
- o Routing determines which segments of the shared data a request must access, and passes the request to those segments, as well as returns the results of the requests to the appropriate processing

function. Alternate routing may be used if more than one copy of the data exists.

- o <u>Sequencing</u> determines the priority of requests to ensure that response times for data requests are within acceptable limits.
- o <u>Concurrency Control</u> ensures that multiple, active requests do not alter the same data so as to create an inconsistent state within the Shared Data Resource.

2.4.7 Shared Data Resources

Shared Data Resource is the component responsible for holding the information common to one or more other components of the Model. Although this activity is logically centralized in the Shared Data Resource, it may contain multiple elements (storing different segments of the shared data, or different organizations of the shared data). The Shared Data Resource performs two functions:

- o <u>Information Management</u> determines what information must actually be stored and retrieved to satisfy the request, performs the transformations necessary to produce the required information, and determines how the information is to be stored or retrieved.
- o <u>Storage Management</u> determines physical locations of data and access storage storage devices.

3. IMPLEMENTATION EXPERIENCE

3.1 SUMMARY OF RESULTS

The conceptual model of Figure 2 has provided general organizational guidelines for system development over the last three years. Two particular projects completed in that time, including two large databases (20 gigabytes and 1.5 gigabytes), will be described. VAXCLUSTER technology and the ORACLE relational database management system was used extensively to implement the conceptual model. A portion of the development of one of the applications used the STRATEGIM language. Most of the other applications were either programmed in the C language or used existing packages.

The goals which lead to the development of these two systems were threefold (in order of importance): 1) to create effective <u>applications</u> for particular user groups; 2) to provide a <u>central repository</u> and further dispatch point for all banking transaction processing historical data; and 3) to provide an application <u>systems architecture</u> in which MIS data of various kinds was organized according to a partial order of increasing levels of abstraction or aggregation, so that higher levels of data would be created by batched flows of data from lower levels.

The design and implementation of the systems to meet these goals were greatly affected by the cultural factors and business considerations mentioned earlier. Furthermore, experiences with the functionality and performance characteristics of ORACLE are also reported along the way.

The first and last of the goals (good applications and structured aggregation) have been largely achieved by the implementation. The second goal (shared historical data resource) was approached very cautiously. Technology makes this goal today more feasible than it was in 1984.

Despite much skepticism about the performance of a relational database system on databases as large and active as these, particularly on minicomputer technology, the applications making use of these databases ultimately performed very well.

3.2 SYSTEM DESIGN

3.2.1 Application of the Conceptual Model

General plans for new systems called for the gradual segregation of systems development efforts, and of hardware and software components, into specific classes of systems which would correspond to components of the conceptual model.

It was important that this be done on an evolutionary basis since the inventory of existing systems constitutes some 20 million lines of code. There was (and is) insufficient business motivation to replace all these working systems even to achieve such general goals as data integration. Instead, as new applications are built and old ones replaced, it is intended that they be brought into greater conformity with the model, on an application-by-application basis. This means that only those pieces of integrative components required to support a developing application may possibly be built.

In addition, it was not practical that the integrative software and hardware required for Message Control and Data Control be custom built for the bank. Instead, it was expected that these components would be purchased commercially. At the time the model was proposed and accepted as the basis for new development in the bank, complete Message Control and Data Control software was not commercially available. Over time, it has increasingly become available. Thus, one of the major values of the model has been positioning the bank for the arrival of such new technology.

The following two sections describe the ways in which the ideas of the conceptual model were applied to the development of the historical database system for transaction investigation and the related MIS system, first from the point of view of their functional organization, and then from the point of view of implementation issues.

3.2.2 Role of a Historical Database

The historical database was envisioned as providing a data resource shared by multiple applications: Transaction processing systems would no longer have the responsibility of storing historical data nor need to produce a variety of different summaries and views of the same data for the different information processing systems. Instead, all completed transactions and proofed account balances of each type would simply be written to a common database. Information processing systems would each extract the data they needed from the historical database, and use that data independently from the uses put to it by other systems.

The goals of such a historical database was to simplify the work of transaction processing systems, and more importantly, to simplify inter-system data flows. The current flows had reached a level of complexity which were, in total, no longer known to any one person and had no discoverable principle of organization. They required several people several months to catalogue. Of most urgency, dependencies on tape hands-off between systems meant that each year more systems were unable to complete their off-line work in the daily time periods available, especially because customers all over the world demanded that some systems be available on-line virtually all the time. The

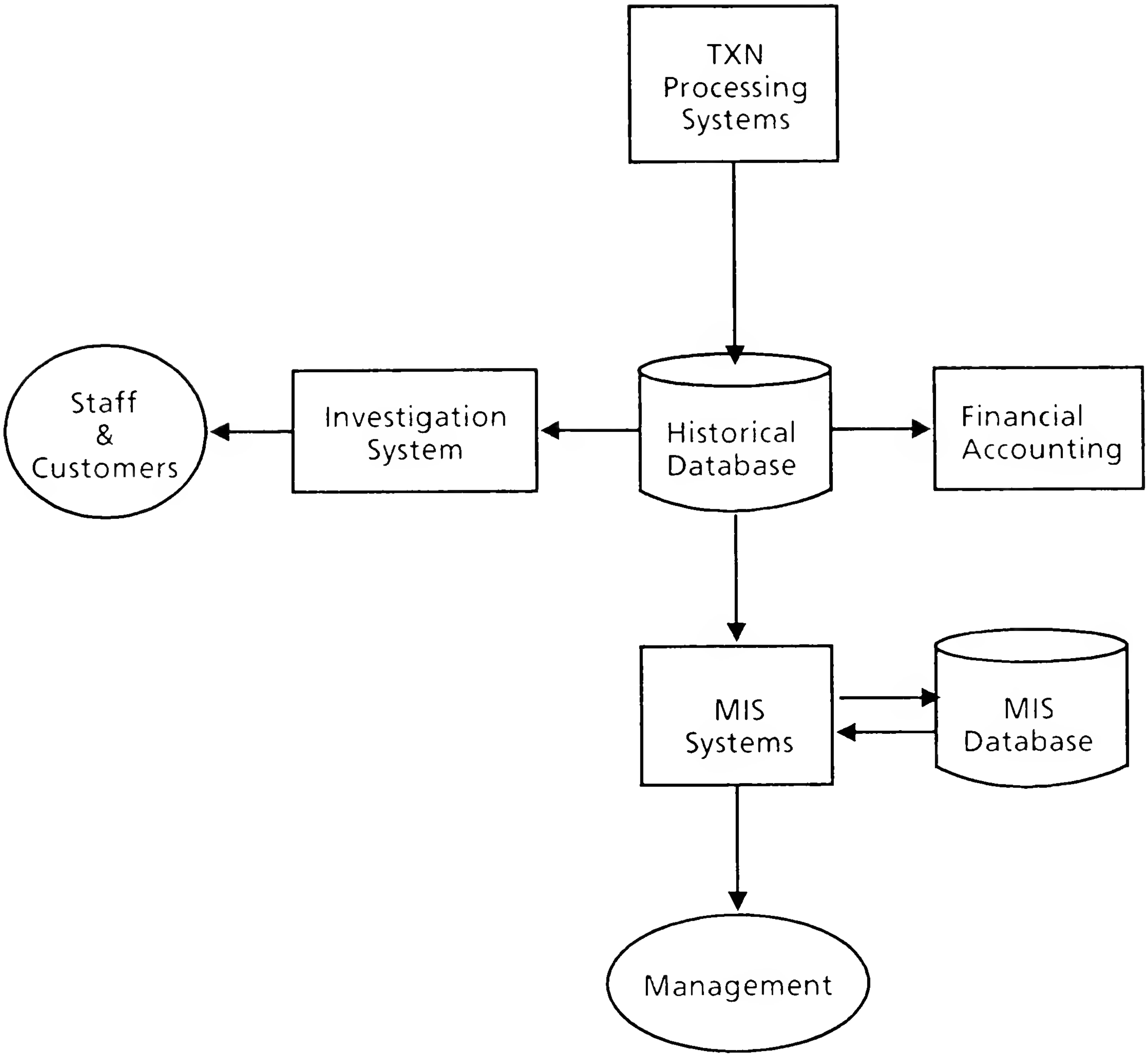


Figure 3.
Role of Historical Database

data-flows to be provide by this new organization are shown in Figure 3.

3.2.3 Structured Aggregation in MIS Systems

The integrated MIS system, in addition to providing the functional benefits discussed above, was, like the transaction investigation system, intended to improve the overall organization of inter-system communications, by deciding in a systematic way which MIS applications should communicate with which other applications.

This was to be accomplished by describing the information input requirements and the information output of various desired MIS applications in a uniform way, and then identifying the lowest cost (least transformation required) connections between such applications, so that the output of some applications becomes the input to others.

Treating the output of each MIS application as a database in its own right, and regarding the primary role of this sort of MIS application (standard cost accounting) as a data aggregation function, the result is a partial order of aggregated databases in which all databases depend ultimately on the raw historical data. Figure 4 depicts the notion of a partial order. The current collection of MIS databases created consists of seven levels and the dependency diagram barely fits on a large wall.

The general design of the MIS system is in fact to regard various sets of tables as levels of aggregation, created by specific applications and supplying data for further applications. Although the processing is quite complex, the fact that all of the data is maintained as a single shared data resource has dramatically simplified the operations of the MIS system components.

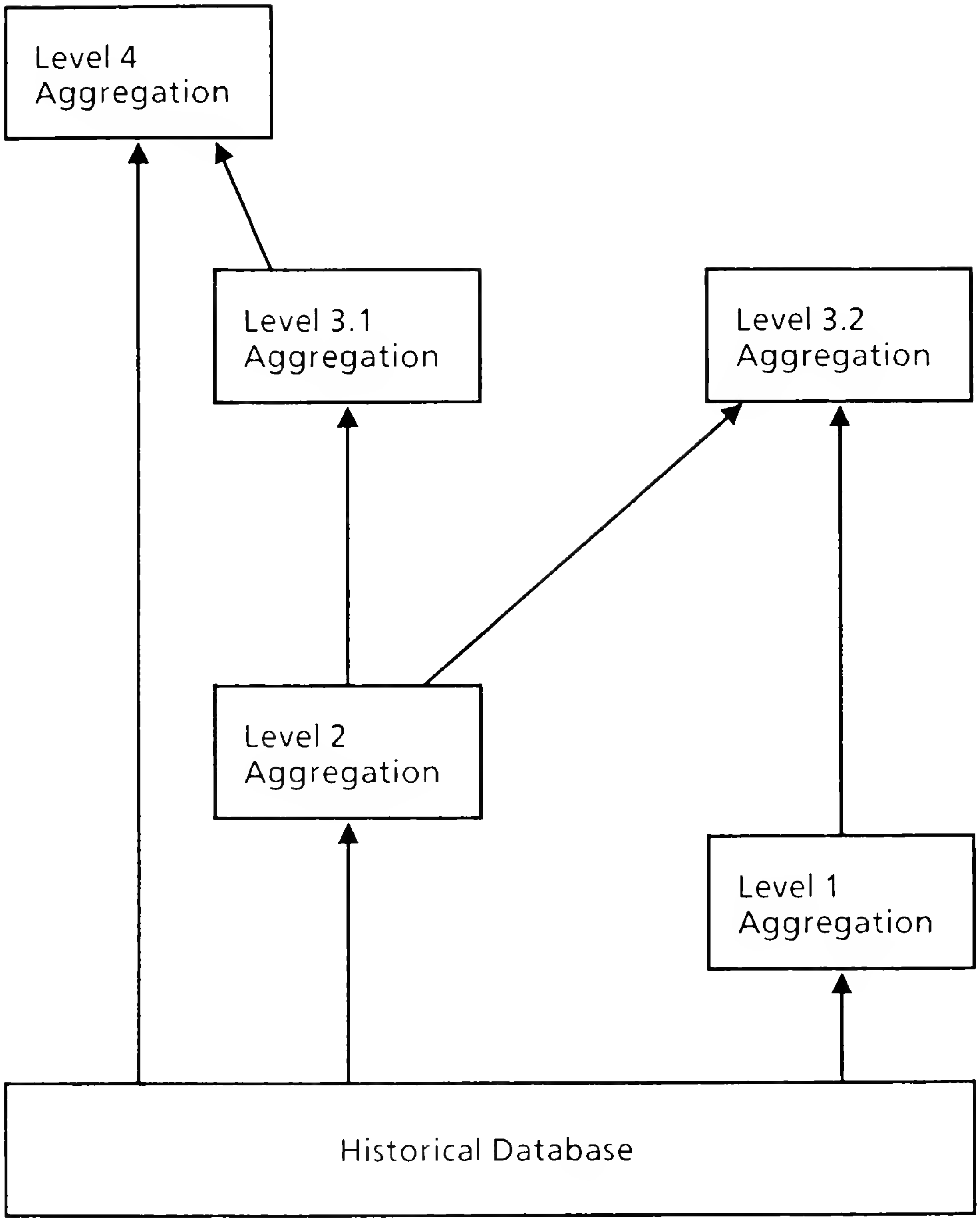


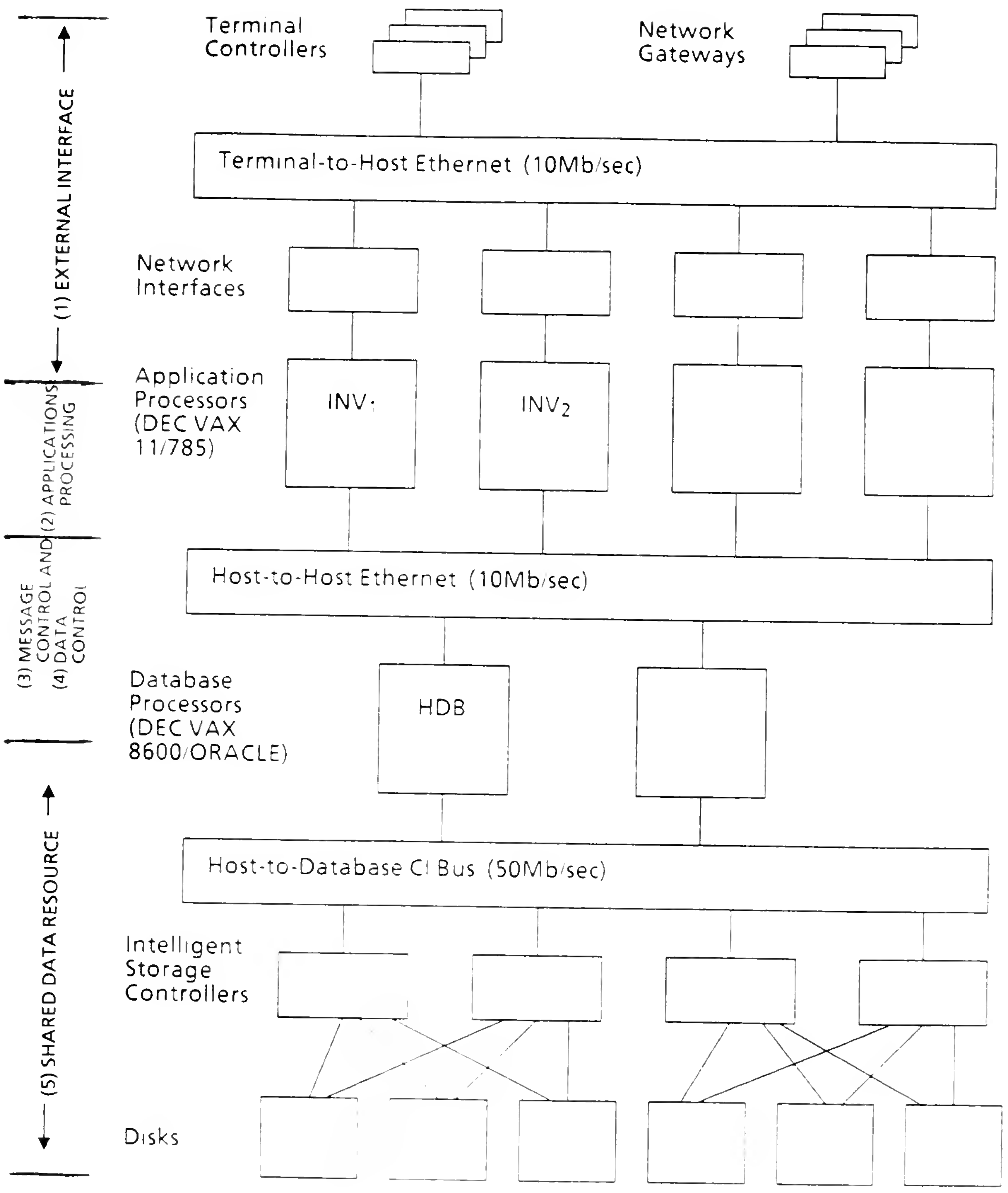
Figure 4.
MIS Data Partial Order of Aggregation

3.3 MODEL INTERPRETATION

By "interpretation" we mean a functional mapping of model components to hardware and software components. The interpretation of the model has varied considerably between the transaction processing and information processing environments, and has varied over time. For instance, front-end processors are now being distributed geographically, which has caused some changes and extensions.

The interpretation of the model presented here was used to provide the environment for the development of the transaction investigation system, and its later interface to the funds transfer system. The initial interpretation involved a "null" message control layer, since the early information processing applications did not communicate with each other, and all screen management was done in the same processor as the applications, although with separate software. Only later, when screen management software running on personal computers and capable of communicating via datagrams with the applications processors became commercially available, did a message control layer become significant in information processing.

The five layers of the conceptual model are interpreted as shown in Figure 5: (1) an external interface system consisting of terminal controllers, wide area gateway boxes, a terminal to host ethernet, network interfaces for the application host machines, and the screen management software on these hosts; (2) an application layer consists only of the application software; (3) message control, sharing a host-to-host Ethernet with data control; (4) data control consisting of software on the application hosts enabling communication with the database processors over the host-to-host Ethernet, the communications



(1) EXTERNAL INTERFACE
(2) APPLICATIONS PROCESSING
(3) MESSAGE CONTROL AND
(4) DATA CONTROL
(5) SHARED DATA RESOURCE
Terminal Controllers
Network Gateways
Terminal-to-Host Ethernet (10Mb/sec)
Network Interfaces
Application Processors (DEC VAX 11/785)
INV1
INV2
Host-to-Host Ethernet (10Mb/sec)
Database Processors (DEC VAX 8600/ORACLE)
HDB
Host-to-Database CI Bus (50Mb/sec)
Intelligent Storage Controllers
Disks

Figure 5
Model Interpretation

software on the front ends of the database processors, and a portion of the database management system, which includes query analysis and concurrency control; and (5) shared data resources consisting of the back ends of the database management systems running on the database processors, and the storage systems to which they are connected. These components are described in more detail below.

3.4 ENVIRONMENTAL AND SYSTEMS SOFTWARE REQUIREMENTS

The bank's development policies emphasize individual projects that are as small as possible, both in number of personnel and time frame. Ideally, projects should take less than a year and require no more than 5 developers [Appleton, 1986].

The environmental software chosen to support these information processing applications differed from that chosen later to support transaction processing applications (such as the funds transfer system). The major reasons for this difference are the technical and performance differences between the transaction processing and information processing systems.

The information processing systems communicate with the outside world largely interactively; their databases are the only components which must be fully recoverable, and the tolerable time to recovery may be as long as several hours. (While for transaction processing systems this is typically a matter of minutes or even seconds). In addition, information processing systems are highly fluid: requirements change from month to month.

The primary environmental software required on the information processing side were flexible screen managers and database management systems, as well as host-to-host communications software. In addition, these systems lend themselves to prototyping and non-procedural languages.

3.4.1 Operating Environment

The hardware and software chosen to realize these components was overwhelmingly based on the DEC VAX. The group's processors currently include about thirty VAXes and two IBM mainframes. While the IBMs exchange messages with the VAX systems, the two environments are not currently integrated into the same conceptual model: the IBM systems, developed earlier, perform high volume batch work running purchased packages which provide end-to-end support for these applications.

The reasons for this preponderance of VAXes were the experience base of available developers and support personnel as well as the cultural forces, noted earlier, favoring separate computers for separate jobs. Thus in cases where preliminary analysis indicated that VAX systems were capable of handling projected workloads, and where specialized software was not immediately available off the shelf for MVS systems, VAXes have invariably been chosen.

In the case of the transaction investigation system, the availability on the VAX of a software package to support investigation tracking was the most important factor in the selection of the VAX, since here the size of the database might have indicated that a mainframe would be a better prima facia choice.

As a result of this choice, VAXCLUSTER technology has been very widely used in a variety of ways. To keep Figure 5 simple, certain complexities have been omitted. For example, some of the applications processors are also connected to the Computer Inter-connect (CI) Bus to allow them to take the place of a defective database processor. Furthermore, for increased reliability, each Ethernet shown has an additional backup Ethernet.

3.4.2 External Interface

All communications with the ultimate system users is terminal based. Certain data are downloaded to PCs for spreadsheet manipulation, graphics, and report printing. Terminal-to-host and PC-to-host communications (part of the External Interface system) is accomplished physically by an Ethernet (In fact, two: one for production systems and another for development systems which serves as backup for the production network).

Bridge controllers (local terminal controllers, dial up-systems, and X.25 gateway controllers, which provides a connection to remote terminals via an in-house worldwide network) are used to serve both the terminals and the computers. They communicate with each other via XNS. Sun Workstations are used on this network as network control and configuration management devises.

Screens are managed from the VAX application machines, using Viking Forms Management software, which in turn communicates with the application software.

3.4.3 Applications

The transaction investigation application software largely consists of calls to the screen management and the database management systems. In addition, it performs any processing required to make the necessary transformations between these two systems. The package purchased to support these investigations, in actuality, does considerably more than this.

3.4.4 Data Control

The Data Control level consists largely of database management software and general purpose hardware. All the features of Data Control could not be implemented, owing to the lack of commercial distributed database management systems at the time of implementation.

In addition to development productivity, the major issues concerned performance, because of the large size of the databases. Other important criteria were: 1) preferences for relational systems as having the longest future and offering the greatest productivity gains, 2) relatively widely used systems, and 3) adherence to standards and ad-hoc standards when possible, such as CODASYL or SQL.

At the start of the project, there was considerable, mostly unsolicited, expression of opinion and concern that relational systems "performed poorly" and were therefore unsuitable for large databases. Of course it is meaningless to talk about good or bad performance except with respect to a specific use and to specific parameters of performance. In this case, both the historical database and the MIS database would be written almost solely in batch mode. Although the volume of such writes was quite large (300,000 512 byte rows per night), the writes were all appends, no update; reads and writes were never expected simultaneously against the same tables; all reads were predictable since very little access to the databases would be ad-hoc; and the database could be designed in such a way that most read requests would require no joins between tables.

Performance projections were established by study of the results of benchmarks performed by other institutions, by review of published performance analyses, including a study by the National Bureau of

Standards, and by analysis of the likely effects of design features of the systems.

It was quickly established that the commercially available systems based on pointer chains (which happened to be coextensive with the available CODASYL systems) were incapable of performing the nightly number of batch appends even within a period of 24 hours, while the systems based on indexes (the relational systems under consideration) would permit these batch appends within a number of hours.

The two systems considered most seriously were INGRES and ORACLE. Benchmarks as well as general opinion suggested that ORACLE's performance in performing unlocked reads against single tables in large databases was significantly superior to that of INGRES. ORACLE exhibited linear (and almost flat) increase in response times to such simple queries as the size of the database increased. INGRES was superior to ORACLE in the performance of complex joins, and offered superior fourth generation development tools.

For the type of production applications envisioned, ORACLE appeared to be a better choice, although neither system would be, in its then current version for VAX hardware, capable of handling the number of simultaneous users ultimately projected for the investigation system. Special design approaches were used to overcome these problems. Furthermore, one could be confident that both hardware and software would get even faster soon.

3.4.5 Shared Data Resource

The data resource used in all the VAXCLUSTER systems is supported by DEC cluster storage, which consists of the very high speed CI Bus

(70 megabits per second), connecting a number of processors and a number of intelligent controllers (currently a total of 16 such devices) in such a way that any processor can communicate with any controller. Controllers are in turn connected in pairs to dual ported disks, allowing full redundancy of hardware components. For instance, within the cluster, when a processor fails, its work can be taken over by another processor which has immediate access to all of the same disks.

3.5 DEVELOPMENT HISTORIES

3.5.1 Changes in the Plans

In developing these systems, it became clear that the strength of certain features of the bank culture was even greater than anticipated: independence and competitiveness between managers, and a predilection for achieving fast tangible results. Steps to integration actually needed to be smaller than the original plans called for. In building and supporting stand-alone applications, each development group has the total responsibility for delivering an application to users, and each group has a strong aversion to relying on cooperation from other groups, not equally responsible for some deliverable.

Only a portion of the transaction processing data (that involved with funds transfers) was required to support the needs of the investigation application. The MIS system, however, required summary data from all the transaction processing systems. These systems already had the capability of providing the lowest level of aggregation required by the MIS applications, since they were supplying this data to current applications. A coordinated effort would therefore provide theoretical future benefits, while increasing current development time, cost, and risk.

As a result, the transaction investigation / HDB and MIS projects, which were started at approximately the same time, wound up each going its separate way. That is, instead of extracting and aggregating data from the HDB, the MIS system receives its own partially aggregated data feeds from transaction processing and financial accounting systems, as shown in Figure 6.

3.5.2 Transaction Investigation System (HDB)

Projections called for at least 100 simultaneous users using the investigation system and having read only access to the historical database, with maximum rates of 1 database request each five seconds.

The applications and the database are supported on a VAXCLUSTER, as shown earlier in Figure 5. A VAX 8600 is used to support the database, while the investigation application runs on multiple VAX 11/785 "front end" machines, which communicate with the database machine via DECNET and the VMS mailbox facility. The investigation application constructs a query in the form of a "query type" and a set of parameters. The historical database system translates this into an SQL query, and returns the response table to the application.

The configuration has grown to the two front-end 11/785s and the 8600 "data base processor" shown in Figure 5, while ORACLE has gone through two major new releases. Several changes in design to improve performance have also been made over the years. The modular evolutionary capability of the conceptual model architecture greatly facilitated all of these changes.

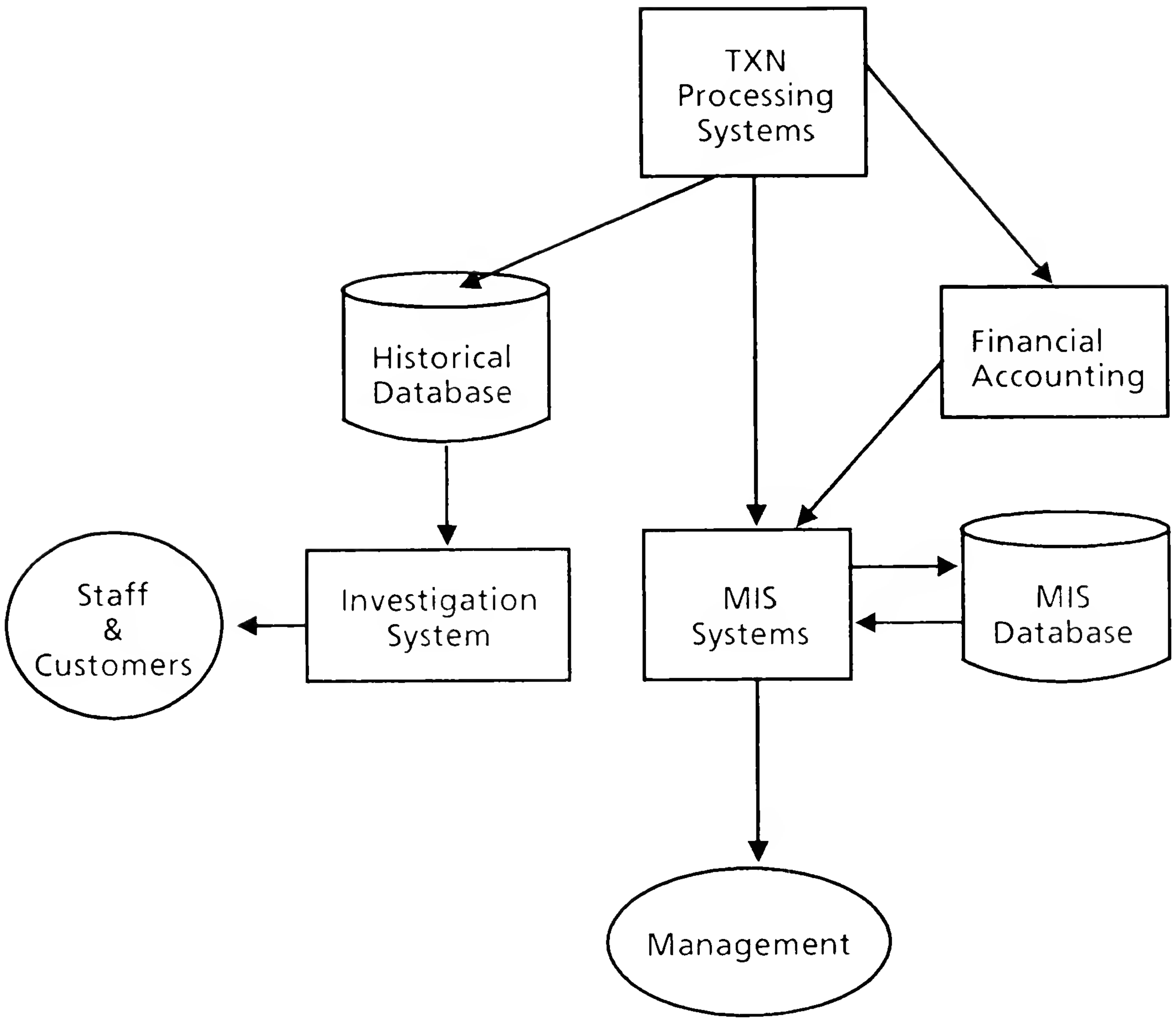


Figure 6.
Current Data Flows

The initial installed production system (on one VAX 11/780 front end and one VAX 11/785 back end) supported 30 users with database query response times of 90 to 120 seconds, meaning that the decoupling of queries and interaction was essential to the acceptability of the system. The current software and hardware supports 160 simultaneous users with database response times of 5 to 7 seconds. Before automation, each search for historical data required at least 15 minutes of an investigator's time.

The full cycle of design, development, testing, installation, training, and live operation of the transaction investigation system required about six months, with one programmer responsible for the software development.

A significant number of small problems, and a few large ones, were discovered in the course of development. Many of these problems could have been anticipated by careful up-front studies. However, such studies would have required as much time as building the system did. Without the flexible and powerful environment used, the same sorts of unanticipated problems would have been likely to cause development disasters, as they often had done in the past in the experimental culture of the bank.

For example, it was discovered during the course of development that the ORACLE "money" data type would not hold large enough amounts for the needs of a very large bank. It was also discovered that the tape recovery of very large tables would take much too long, and that mirroring was not economically justifiable, while at the same time, the re-indexing of very large tables was taking too long (the original analysis counted only the time required to index the newly appended data each day, not the entire 20 gigabyte database.) To solve this problem, tables were partitioned by dates, and application code was written to permit searches across ranges of dates.

3.5.3 MIS Systems

The initial phase of MIS system development and implementation also required about six months, with three contract developers and one manager. About one third of this time was devoted to learning to read correctly the tapes from the various non-integrated sources of MIS data, and developing programs to map these tapes to relational tables without repeating groups, multiple record types, variable length records, etc.

A major innovation of the system was the virtual elimination of the printing of MIS reports: particular pages of reports of interest to individual managers are viewed on terminals, and printed locally if so desired.

The MIS system had intense advocates in the MIS department, who specified exactly what they wanted to see in the system. It is more common in the bank for users to ignore new systems until they are delivered to them. The system has been constantly enhanced and enriched, with the MIS group developing a great deal of skill at the use of SQL and ORACLE development tools for small applications. In fact, this system is not called the MIS system by most of the users, it is called the ORACLE system.

The summary data from this system began to be used in business planning sessions, which lead to the desire to use the data for forecasting and analytic modelling purposes. The result was the creation of a strategic MIS system, using the STRATAGEM modelling language, which receives data from some of the highest level applications of the ORACLE MIS system. The use of this system for analytic and strategic purposes, rather than purely reporting purposes, has grown slowly but steadily.

4. CONCLUDING REMARKS

The conceptual model described in this paper has served as an evolutionary blueprint. The movement from complete autonomy to complete integration is slow and will probably never occur in this organization, nor any organization realistically.

By separating the external interfaces, message control, data control, and the database components from the application processing components, the approach presented here provides for high degrees of integration while preserving significant autonomy - and the ability to evolve further in both directions.

Acknowledgements

Work reported herein has been supported, in part, by the Department of Transportation's Transportation System Center, the Air Force, the Space and Naval Warfare Systems Command, the Rome Air Development Center, and Citibank.

5. REFERENCES

1. Appleton, D.S., "Very Large Projects," Datamation, January 1986.

2. Hsiao, D. K. and Madnick, S. E., "Database Machine Architecture in the Context of Information Technology Evolution," Proceedings of the Third International Conference on VLDB, pp. 63-84, October 6-8, 1977.

3. Keen, P.W., Competing In Time: Using Telecommunications for Competitive Advantage, Ballinger, 1986.

4. Lam, C.Y. and Madnick, S.E., Composite Information Systems - A New Concept in Information Systems," CISR Working Paper # 35, Sloan School of Management, MIT, 1981.

5. Lipis, A., Electronic Banking, John Wiley and Sons, 1985.

6. McFarlan, F.W., "Information Technology Changes the Way You Compete," Harvard Business Review, May-June 1984, pp. 98-103.

7. Madnick, S. E., "Trends in Computers and Computing: The Information Utility," Science, Vol. 185, March 1977, pp. 1191-1199.

8. Madnick, S.E. and Wang, Y.R. "Evolution Towards Strategic Applications of Very Large Data Bases Through Composite Information Systems," Working Paper #1862-87, Sloan School of Management, MIT, February 1987.

4953 058